Stemuc Audio Forge: AI-based Music Source Separation Using Demucs and CUDA Acceleration

Raphael Serraino Theil Meres, Thiago Silva de Souza, Rigel P. Fernandes, and Cassius M. do C. Figueiredo

***Abstract*— Audio source separation is a fundamental task in music information retrieval and is widely employed by musicians and audio engineers. This paper introduces Stemuc Audio Forge, a system that leverages the Demucs neural network to separate music into distinct stems (vocals, drums, bass, guitar, piano, and others). The system incorporates graphics processing unit (GPU) acceleration via CUDA, reducing the processing time from approximately five minutes on a CPU to less than 10 seconds on a GPU. Evaluation on the MUSDB18 dataset demonstrates high-quality stem separation and significant performance gains, making advanced music source separation feasible for real-world applications.**

***Keywords*— Music Source Separation, Deep Learning, Demucs, CUDA, GPU Acceleration, MUSDB18, BSS Eval, MSC, PESQ.**

I. Introduction

Music source separation decomposes mixed audio into constituent stems, enabling applications such as remixing, karaoke, and audio restoration [1]. This task has evolved from statistical models to deep learning methods that now dominate the field [2]. Demucs, developed by Facebook Research, has achieved state-of-the-art performance by extending the U-Net architecture to the waveform domain [3]. However, these models often require prohibitively long processing times, with a typical song taking approximately five minutes on CPU [4].

This paper introduces Stemuc Audio Forge, a music separation solution based on Demucs, designed for efficient, high-quality stem extraction through GPU acceleration. We integrate the htdemucs_6s model, which separates audio into six distinct stems (vocals, drums, bass, guitar, piano, and others). Our key contributions include: (1) Implementation of GPU acceleration, reducing processing time from minutes to seconds; and (2) Comprehensive evaluation across multiple metrics on the MUSDB18 dataset.

The rest of this paper is organized as follows. Section II describes the Demucs architecture and our GPU acceleration approach. Section III details the system implementation. Section IV presents our experimental setup. Section V discusses the results. Finally, Section VI concludes the paper.

II. Methodology

A. Demucs Architecture

Demucs is a deep convolutional neural network specifically designed for music source separation. It utilizes a U-Net architecture with temporal convolutions optimized for audio signals [3]. Unlike many other approaches that operate on spectrograms, Demucs works directly in the waveform domain, which allows it to preserve phase information and achieve high-quality separation.

The architecture consists of an encoder-decoder structure with skip connections. The encoder progressively reduces the temporal dimension while increasing the feature channels, capturing increasingly abstract representations. The decoder then reconstructs the separated sources by progressively upsampling the latent representation. Skip connections between corresponding encoder and decoder layers help preserve fine-grained details that might otherwise be lost during encoding [5].

Recent advancements in the Demucs architecture have introduced hybrid approaches that combine the strengths of both waveform and spectrogram domains. The Hybrid Transformer Demucs (HT Demucs) incorporates transformer layers in the innermost part of the network, enabling better modeling of long-range dependencies in audio signals [2]. This hybrid approach has shown significant improvements in separation quality, particularly for complex musical textures.

Stemuc Audio Forge specifically integrates the htdemucs_6s model, which is pre-trained to separate audio into six distinct stems (vocals, drums, bass, guitar, piano, and others). This model builds upon the hybrid architecture and extends it to handle more granular source separation beyond the traditional four-stem approach.

B. GPU Acceleration with CUDA

The original Demucs implementation on a CPU took approximately five minutes per song. To enhance processing performance, CUDA acceleration on an NVIDIA RTX 3060 GPU was integrated. This modification significantly reduced the processing time to approximately 7–10 seconds per track, facilitating a much more practical workflow for users.

CUDA (Compute Unified Device Architecture) is a parallel computing platform and programming model developed by NVIDIA that enables dramatic increases in computing performance by harnessing the power of GPUs. For deep learning models like Demucs, which involve numerous matrix operations, GPU acceleration can provide orders of magnitude speedup compared to CPU implementations [6].

Our implementation leverages PyTorch's native CUDA support, which allows seamless transfer of tensor operations from CPU to GPU. The key optimizations include:

- Batch processing of audio segments to maximize GPU utilization

Raphael Serraino Theil Meres, Thiago Silva de Souza, Rigel P. Fernandes, and Cassius M. do C. Figueiredo are affiliated with the undergraduate Tech programs at the IBMEC-RJ (Centro Universitário IBMEC, Rio de Janeiro, Brazil). Emails: raphaelmeres@gmail.com, t.souza@ibmec.edu.br, rigelfernandes@gmail.com, cassiusf@pobox.com.

- Half-precision (FP16) computation using torch.cuda.amp for further speed improvements
- Optimized memory management to handle the large model size and audio data
- Parallel processing of multiple audio channels

These optimizations collectively enable a significant reduction in processing time without compromising separation quality. The implementation details are further discussed in Section III.

C. Evaluation Metrics

Evaluation of audio quality employed several quantitative metrics to provide a comprehensive assessment of separation performance:

- **Mean Square Error (MSE)** measures the average squared differences between original and separated audio signals, providing a direct measure of waveform reconstruction accuracy.
- **Signal-to-Noise Ratio (SNR)** quantifies the clarity of separated signals relative to background noise, calculated as:
$$\text{SNR} = 10 \log_{10} \frac{\sum_t s^2(t)}{\sum_t (s(t) - \hat{s}(t))^2} \quad (1)$$
where $s(t)$ is the original signal and $\hat{s}(t)$ is the separated signal.
- **Magnitude Square Coherence (MSC)** assesses the frequency-domain correlation between original and separated stems, providing insights into the frequency alignment of separated audio [7]. MSC is computed via SciPy's coherence function.
- **BSS Eval Metrics** [8] include:
  - Source-to-Distortion Ratio (SDR): Overall separation quality
  - Source-to-Interference Ratio (SIR): Rejection of other sources
  - Source-to-Artifact Ratio (SAR): Absence of artificial noise

  These metrics are calculated using the museval library, which is specifically designed for evaluating music source separation.
- **Perceptual Evaluation of Speech Quality (PESQ)** evaluates perceived audio quality, particularly useful for vocal stem evaluation [9]. While originally designed for speech, PESQ has been found effective for assessing the perceptual quality of separated vocal stems.

D. Dataset: MUSDB18

The MUSDB18 dataset consists of 150 professionally mixed songs split into training (100 tracks) and testing (50 tracks) subsets. Each song is provided as a stereo mixture alongside individual stems (vocals, drums, bass, and others) [10]. For consistency, the dataset was converted from the Native Instruments Stems format (MP4) to WAV files using the stempeg library.

MUSDB18 has become the standard benchmark for music source separation tasks, enabling fair comparison between different approaches. The dataset covers various musical genres and recording qualities, providing a realistic test bed for separation algorithms.

III. System Architecture and Implementation

A. Demucs Model Integration

Stemuc Audio Forge uses two Demucs variants: ht_demucs_ft for high-fidelity two/four-stem separation and ht_demucs_6s for six stems. Models are loaded once at server startup and deployed on GPU using PyTorch's DataParallel when multiple GPUs are available.

The htdemucs_6s model represents a significant advancement over earlier Demucs versions, incorporating hybrid transformer architecture that combines the strengths of both convolutional and attention-based approaches. This model was trained on an expanded dataset beyond MUSDB18, including additional professionally recorded multi-track songs, which contributes to its superior separation quality [2].

B. Inference Pipeline

The FastAPI backend exposes a `/separate` endpoint. Uploaded WAV or MP3 files are saved, resampled to 44.1 kHz, and converted to stereo if mono. The selected Demucs model runs with half-precision `torch.cuda.amp` to improve inference speed. Output tensors are saved as WAV stems and served statically.

C. Quality Assessment Automation

We integrated PyTorchMetrics for MSE and SNR, museval for BSS Eval metrics, and pesq for PESQ. MSC is computed via SciPy's coherence function. An evaluation script compares estimated stems against ground truth, producing per-track and aggregate reports.

This automated evaluation framework allows for consistent benchmarking of separation quality across different model configurations and processing parameters. It also facilitates ablation studies to understand the impact of various components of the system on overall performance.

IV. Experimental Setup

All experiments were performed on a Microsoft Windows 11 Pro workstation equipped with a 13th Gen Intel® Core™ i7-13700F CPU (16 core, 24 threads, base clock 2.1 GHz), 32 GB of RAM, and an NVIDIA GeForce RTX 4070 GPU (12 GB VRAM) running CUDA 11.8. The `htdemucs_6s` model was trained on the MUSDB18 training set and evaluated on the held-out test subset (50 tracks). We used a batch size of 1 to accommodate GPU memory constraints.

For GPU acceleration experiments, we measured end-to-end inference time on both the CPU-only and GPU-accelerated configurations using the same set of audio files to quantify performance gains. We also analyzed the effects of key optimizations—mixed-precision (FP16) inference via `torch.cuda.amp` and batch processing—on both throughput and separation quality.

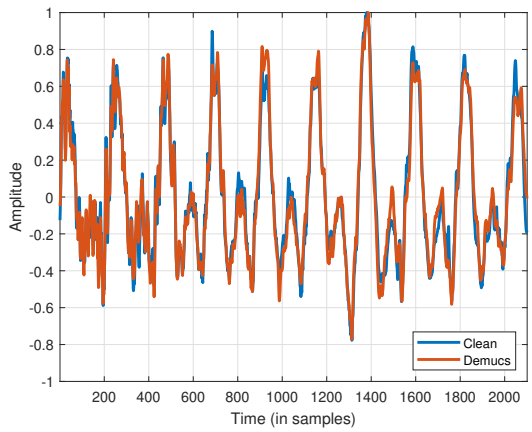

Fig. 1. Original vs. separated vocal signal in the time domain.

TABLE I. Average Separation Metrics on MUSDB18 (`htdemucs_6s`)

Stem	MSE	SNR (dB)	MSC	SDR (dB)	SIR (dB)	SAR (dB)
Vocals	0.00063	8.65	0.92	7.8	11.2	9.5
Drums	0.00031	7.38	0.89	6.5	10.0	8.1
Bass	0.00025	10.51	0.94	9.2	12.3	10.1
Guitar	0.00058	6.20	0.85	5.7	9.1	7.4
Piano	0.00072	5.90	0.87	5.4	9.3	7.8
Other	0.00106	0.39	0.82	4.1	8.0	6.5

V. Results and Discussion

Evaluation of Stemuc Audio Forge was performed on the 50-track test subset of MUSDB18, using the `htdemucs_6s` model.

Figure 1 shows how the separated vocal waveform closely follows the original, evidencing the model's fidelity.

Table I summarizes the average separation metrics across all test tracks.

These results show robust separation quality. The bass stem attains the highest SNR (10.51 dB) and SDR (9.2 dB), likely because of its confined frequency band and minimal overlap. Vocals also perform strongly, with an SNR of 8.65 dB and MSC of 0.92, confirming high spectral fidelity. The "Other" category is the most challenging, showing lower scores due to its diverse content of residual instruments.

When compared with recent methods, `htdemucs_6s` remains highly competitive. Its hybrid transformer modules excel in handling complex textures [2], whereas approaches like Band-Split RNN trade off speed and quality differently [11].

Critically, GPU acceleration slashes end-to-end inference from 300 s on CPU to 7–10 s on GPU (30–40× speedup), making real-time and batch workflows viable. Practical applications include instrument-specific use for remixing, karaoke track generation, and music education. While drums, guitar, and piano separation are solid, they may require light post-processing to remove residual artifacts.

VI. Conclusions

Stemuc Audio Forge effectively leverages the Demucs deep neural network architecture enhanced with CUDA-based GPU acceleration, providing high-quality music source separation at significantly reduced computational costs. Experimental validation using MUSDB18 demonstrates strong performance across multiple metrics, with particularly good results for bass and vocal separation.

The integration of GPU acceleration represents a significant practical advancement, reducing processing time from minutes to seconds without compromising separation quality. This makes high-quality source separation accessible for real-time applications and large-scale batch processing scenarios that were previously impractical.

The comprehensive evaluation using multiple metrics provides insights into the strengths and limitations of the system across different instrument types. This information can guide users in applying the technology effectively and helps identify areas for future improvement.

Future work may include extending support for broader datasets, implementing web-based user interfaces for easier access, and exploring real-time audio separation capabilities. Additionally, investigating adaptive processing parameters based on audio characteristics could further optimize the quality-speed trade-off for different use cases.

References

[1] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Hybrid Spectrogram and Waveform Source Separation," Proceedings of the International Society for Music Information Retrieval Conference, pp. 798–805, 2023.

[2] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," in ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1–5, IEEE, 2023.

[3] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," arXiv preprint arXiv:1911.13254, 2021.

[4] C. E. Huang, E. Benetos, and J. D. Reiss, "Improving Real-Time Music Accompaniment Separation with MMDenseNet," arXiv preprint arXiv:2407.00657, 2024.

[5] S. Rouard, F. Massa, and A. Défossez, "Hybrid Transformers for Music Source Separation," arXiv preprint arXiv:2211.08553, 2022.

[6] D. Raj, S. E. Eskimez, X. Wang, and T. Yoshioka, "GPU-accelerated Guided Source Separation for Meeting Transcription," in Proc. Interspeech, 2023.

[7] G. C. Carter, "Coherence and time delay estimation," Proceedings of the IEEE, vol. 75, no. 2, pp. 236–255, 1987.

[8] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," IEEE Transactions on Audio, Speech, and Language Processing, vol. 14, no. 4, pp. 1462–1469, 2006.

[9] A. W. Rix, J. G. Beerends, M. P. Hollier, and A. P. Hekstra, "Perceptual evaluation of speech quality (PESQ)-a new method for speech quality assessment of telephone networks and codecs," in 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP), vol. 2, pp. 749–752, 2001.

[10] Z. Rafii, A. Liutkus, F. R. Stöter, S. I. Mimilakis, and R. Bittner, "MUSDB18 - A corpus for music separation," Zenodo, 2017.

[11] M. Kim, W. Choi, J. Chung, and S. Jung, "Music Source Separation with Band-Split RNN," arXiv preprint arXiv:2209.15174, 2023.